

Objective 1 **Get to Know the Command Shells**

You cannot communicate directly with the operating system kernel. You need to use a program that serves as an interface between the user and operating system.

In the operating systems of the UNIX family, this program is called the *shell*.

The shell accepts a user's entries, interprets them, converts them to system calls, and delivers system messages back to the user, making it a command interpreter.

To understand command shells, you need to know the following:

- [Types of Shells](#)
- [bash Configuration Files](#)
- [Completion of Commands and Filenames](#)

Types of Shells

UNIX has a whole series of shells, most of which are provided by Linux in freely usable versions. The following are examples of some popular shells:

- The Bourne shell (**/bin/sh**; symbolic link to **/bin/bash**)
- The Bourne Again shell (**/bin/bash**)
- The Korn shell (**/bin/ksh**)
- The C shell (**/bin/csh**; symbolic link to **/bin/tcsh**)
- The TC shell (**/bin/tcsh**)

The various shells differ in the functionality they provide.

Every shell can be started like a program and you can switch at any time to a different shell. For example, you can switch to the C shell by entering **tcsh**; you can switch to the Korn shell by entering **ksh**.

Unlike most other programs, the shell does not terminate on its own. You need to enter the **exit** command to return to the previous shell.

A shell is started at a text console right after a user logs in. This is called the *login shell*. Which shell is started for which user is determined in the user database.

The standard Linux shell is bash, so we will only cover bash shell in this objective.

bash Configuration Files

To customize bash for an interactive session, you need to know about the configuration files and about the order in which they are processed.

To understand how shells work, you need to know the difference between the following:

- [Login Shells](#)
- [Non-Login Shells](#)

Login Shells

A login shell is started whenever a user logs in to the system. By contrast, any shell started from within a running shell is a non-login shell. The only differences between these two are the configuration files read when starting the shell.

A login shell is also started whenever a user logs in through an X display manager. Therefore, all subsequent terminal emulation programs run non-login shells.

The following files are read when starting a login shell:

1. **/etc/profile**. A system-wide configuration file read by all shells. It sets global configuration options. This configuration file will be read not only by the bash, but also by other shells.

~/.profile is a file created for each new user by default on the SUSE Linux Enterprise Server. Any user-specific customizations can be stored in it.

2. **/etc/bash.bashrc**. some useful configurations for the bash shell are made. For example:
 - Appearance of the prompt
 - Colors for the **ls** command
 - Aliases

For your own system-wide bash configurations use the **/etc/bash.bashrc.local** file that is imported from **/etc/bash.bashrc**.

~/.bashrc is a configuration file in which users store their customizations.

Non-Login Shells

Only the **/etc/bash.bashrc**, **/etc/bash.bashrc.local** and **~/.bashrc** files are read when a non-login shell is started.

Like most other Linux distributions, SUSE Linux Enterprise Server has a default setup that ensures users do not see any difference between a login shell and a non-login shell. In most cases, this is achieved by also reading the **~/.bashrc** file when a login shell is started.

If you change any settings and want them to be applied during the same shell session, the changed configuration file needs to be read in again.

The proper way to read in a changed configuration file and to apply the changes to the current session is by using the internal shell command **source**, as in the following example:

```
source ~/.bashrc
```

You can also use the “short form” of this command, which happens to be included in many configuration files, where it is used to read in other configuration files, as in the following (with a space between the period and the tilde):

```
. ~/.bashrc
```

Completion of Commands and Filenames

The bash shell supports a function of completing commands and filenames. Just enter the first characters of a command (or a filename) and press **Tab**. The bash shell completes the name of the command.

If there is more than one possibility, the bash shell shows all possibilities when you press the Tab key a second time. This feature makes entering long filenames very easy.

Objective 2 Execute Commands at the Command Line

If you do not have a graphical user interface, you can use the following to help make entering shell commands to administer SUSE Linux Enterprise Server much easier:

- [History Function](#)
- [Switch to User root](#)

History Function

bash stores the commands you enter so you have easy access to them. By default, the commands are written in the **.bash_history** file in the user's home directory. In SUSE Linux Enterprise Server, the size of this file is set to a maximum of 1,000 entries.

You can display the content of the file by using the **history** command.

You can display the commands stored in the history cache (one at a time) by using the arrow keys. **Up-arrow** shows the previous command; the **Down-arrow** shows the next command. After finding the desired command, edit it as needed then execute it by pressing **Enter**.

When browsing the entries of the history, you can also select specific commands. Typing one or several letters, and pressing **PageUp** or **PageDown**, displays the preceding or next command in the history cache beginning with this letter.

If you enter part of the command (not necessarily the beginning of the command), pressing **Ctrl+r** searches the history list for matching commands and displays them. Searching starts with the last command executed.

Switch to User root

If you are working with a shell, you can become root by entering the **su -** command and the root password.

You can check to make sure you are root by entering **id** or **whoami**. To quit the root administrator shell, you enter the **exit** command.

Exercise 6-1 Execute Commands at the Command Line

In this exercise, use the history feature of the shell and get root permissions at the command line. To do this, use the **history** and **su** command.

Detailed Steps to Complete this Exercise:

1. Open a terminal window.
2. View the history cache by entering **history**
3. Press **Up-arrow** until you see an **ls** command you would like to execute; then press Enter.
4. Type **h** and press **Page Up** once.
You should see the **history** command at the command line again.
5. Press Enter to execute the **history** command.
6. Switch to root by entering **su -**. Then enter a password of **novell**.
7. Check to make sure you are logged in as root by entering **id**
8. Start YaST by entering **yast2**
YaST should start in QT mode.
9. Quit YaST by selecting **File > Quit**.
10. Become the user geeko again by entering **exit**
11. Close the terminal window by entering **exit**

(End of Exercise)